UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/071,159	02/07/2002	Paul Marcos	10010.946CPA2C	9761

22804 7590 11/14/2005

THE HECKER LAW GROUP
1925 CENTURY PARK EAST
SUITE 2300
LOS ANGELES, CA 90067

EXAMINER	
BANANKHAH, MAJID A	
ART UNIT	PAPER NUMBER
2195	

DATE MAILED: 11/14/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/071,159	MARCOS ET AL.	
	Examiner	Art Unit	
	Majid A. Banankhah	2195	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *02 September 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-31* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-31* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Terminal disclaimer filed by applicant on September 02, 2005, is acknowledged. Claims 1-31 are considered for examination.

Claim Rejections - 35 USC § 102

> (a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

Claims 1-31 are rejected under 35 U.S.C. 102(a) as being anticipated by CORBA and ORB (The Server side of CORBA, OS/2 developer ORFALI et al. July/August 1995, OMG, "Understanding the ORB", Part 2, 1995, pp. 73-90, IONA Technologies, "The Orbix Architecture", January 1995, pp. 1-23, and Mowbray et al, "THE ESSENTIAL COBRA System Integration Using Distributed Objects", 1995, RYMER, JOHN, "Distributed Object Interoperability", and "OMG's UNO, Distributed Computing Monitor").
For completeness, the above references are made part of the statement of the rejection because, these references all deal with CORBA and ORB and they discuss various aspect of the CORBA.

As per independent claims 1 and 12:

A method of dynamically communicating an object message between a client and server of separate object models comprising the steps of (ORFALI, page 1, CORBA, and what CORBA does on the client side, and Dynamic Invocation Interface (DII) APIs and associated discussion):

mapping said client to said server [ORFALI, page 2, last line to page 3, first two line, Object adaptor assigns references to new object it creates, and associated discussion on

mapping, also Mowbray, page 36, Fig. 3.2, and page 38, Fig. 3.2, and OMG, page 76, ORB accept responsibility, and associated discussion, further, IONA, page 10/23, 5th paragraph, Smart Proxies, "call-backs" from a server with which it had earlier corresponded, and associated discussion];

intercepting a message generated by said client in a first object model [ORFALI, page 2, "Dynamic Skeleton Interface, and Object adaptor", and their associated discussion, also OMG,, page 74, item 1, "The ORB receives a request targeting an object in the server, The ORB checks its repository and determine that neither the server nor the object is currently active", and IONA, 4th paragraph, "instead object requests are passed directly from the client code to the invoked object implementation", and associated discussion, further, Mowbray, page 36, fig. 3.1, "ORB functions as a communication infrastructure", and page 38, Fig. 3.2., also second paragraph, and associated discussion respectively];

examining a second object model for interface information for said Server [ORFALI, PAGE 2, 1st paragraph, "In both case (dynamic or static invocation), the ORB locates a server object adaptor, transmits the parameters, and transfers control to the object implementation through the server IDL stub", also, OMG, page 74, items "2.", and "3.", IONA, page 12/23, "then Orbix binds the client to an object within the server which provides an interface compatible with that expected the server", and finally Mowbray, page 38, second paragraph to page 39, "exception can be generated by the server or by the ORB in case of errors", and associated discussion];

generating a translated message for said server [ORFALI, page 4, item 5, "the proxy object call cooperates with the local Objects Manager to find a server, marshals method

argument, convert object pointers in the clients", and associated discussion; and OMG, page 78, 2nd paragraph, "object references are converted between stringfield and invocable forms only by the ORB", and associated discussion, also, IONA, page 7/23, 6th paragraph, "The IDL compiler is primarily a part of the development environment, and used to translate IDL description into sub code to aid remote operations", further, Mowbray, page 38, 2nd paragraph, Transparently, the stub provides an interface to the ORB that performs marshaling to encode and decode the operation's parameters into communication formats suitable for transmission", and associated discussion].

forwarding said translated message to said server [the forwarding of the translated message to the server is an inherent part of the process after the translation for the server in any of the references, See ORFALI, "send the method to the server", OMG, page 77, 2nd par. "object adaptor provides interface between ORB and object", IONA, page 7/23, e.g. "to aid remote operation", and page 38, "the OMG IDL skeleton program is the corresponding server-aid implementation of the OMG IDL interface"].

As to statement in the preamble that "message between a client and server of separate object model", applicant's attention is directed to the teaching of Mowbray on page 37, where he explicitly says: "**CORBA is a peer-to-peer distributed computing facility where all applications are objects (in the sense of object orientation). Objects can alternate between client roles and server roles. An object is in a client role when it is the originator of an object invocation. An object is in a server role when it is the recipient of an object invocation, most objects probably will play both roles**", and associated discussion, ass also Fig. 3.2. where client object and server object are separate].

As to independent claim 22:

A dynamic object message broker comprising [the teaching of ORFALI et al., OMG, IONA, and Mowbray et al., on CORBA]:

a first computer system having a first object model and a first object running in said first object model [ORFALI et al., page 1, client object, OMG, page 74-75, client object IONA, page 8/23, client object, and Mowbray et al., page 38, Fig. 3.2. client object];

an mediating component coupled to said first computer system, said mediating component capable of creating a dynamic messaging interface [ORFALI, page2, "the object adaptor sits on top of ORB's core communication services and accept requests for service on behalf of the servers object", and page 1, last par., "to support both static and dynamic client/server invocations", OMG, page 77, 2nd par. IONA, page 11/23, Orbix Architecture (see specifically 2nd par.), and Mowbray et al., page 36, and page 38, ORB] ; and
a second computer system coupled to said mediating component, said second computer system having a second object model and a second object running in said second object model [ORFALI et al., page 1, server object, OMG, page 74-75, server object IONA, page 8/23, server object, and Mowbray et al., page 38, Fig. 3.2. server object].

Regarding first computer, second computer, and mediating component and See, RYMER, "Distributed Object Interoperability" page 7, under the Title, "TWO KINDS OF INTERFACE", also, "OMG's UNO", page 3, under the Title "INTER-ORB BRIDGE SUPPORT AND DYNAMIC SKELETON INTERFACE".

As to the limitation of "first object running in said first object model", and "second object running in said second object model", see the discussion of separate object models in the rejection of claim 1, and 12. Additionally, see the discussion of Mowbray on page 44, regarding "product availability" and cross-platform platform development product called "Common Object Model". See also OMG, page 78, where he teaches of "ORBs will typically lack static skeletons for remote invocations, since the installations procedure for an object implementation grafts its skeleton only onto its own ORB and not on that of the client", indicating explicitly that the objects in OMG have their own communication system , also page 87, last par., (See also page 4 of the present application, for the definition of Object Model on page 3 last par. continued on page 4).

As to independent claim 27:
The claim is rejected for the reasons stated in the rejection of claims 1, and 12, and in addition
creating a proxy object associated with said client [ORFALI, page 4, item 5, "in the client by the local proxy objects", also first partial par., classes that provide the proxy client, server, and socket functions, OMG, page 74-75, IONA, page 9/23, 2nd to last par., "runtime builds a "proxy", and Mowbray et al., page 38];

creating a stub object associated with said server [ORFALI, page 1, "and both client and server stub are generated by the IDL compiler", OMG, page 87, IONA, page 9/23, 2nd to last par., "runtime builds a "proxy", and Mowbray et al., page 38];
determining a message protocol for said server [see, Mowbray, page 22, Fig. 1.4., and associated discussion];

As to independent claim 31:

The claim is rejected for the reasons stated in the rejection of claims 1, and 12, and in addition

examining a plurality of second object models to locate a server to process said message [ORFALI, PAGE 2, 1st paragraph, "In both case (dynamic or static invocation), the ORB locates a server object adaptor, transmits the parameters, and transfers control to the object implementation through the server IDL stub", also, OMG, page 74, items "2.", and "3.", IONA, page 12/23, "then Orbix binds the client to an object within the server which provides an interface compatible with that expected the server", and finally Mowbray, page 38, second paragraph to page 39, "exception can be generated by the server or by the ORB in case of errors", and associated discussion, also IONA, page 8/23],

obtaining interface information for said server running in one of said plurality of second object models [ORFALI et al., page 1, last par. "the ORB interface consists of a few APIs", OMG, page 78, "the dynamic skeleton interface" associated discussion, IONA, page 9/23, last par. IDL interface, and Mowbray et al., page 37, Fig. 3.1. interface definition language and Fig. 3.2., and associated discussion];

As to dependent claim 2:
The method of claim 1 further comprising the step of transmitting a response from said server to said client [ORFALI et al., page 2, "The object adaptor", and associated discussion, OMG, page 74, "item No. 5, IONA, page 9/23, 4th par., and Mowbray et al., page 38, Fig. 3.2.].

As to dependent claims 3 and 13:
The method of claim 1 further comprising the steps of;

said client sending a query to determine if said server is able to respond to said message [ORFALI, page 2, "Dynamic Skeleton Interface, and Object adaptor", and their associated discussion, also OMG,, page 74, item 1, "The ORB receives a request targeting an object in the server, The ORB checks its repository and determine that neither the server nor the object is currently active", and IONA, 4th paragraph, "instead object requests are passed directly from the client code to the invoked object implementation", and associated discussion, further, Mowbray, page 36, fig. 3.1, "ORB functions as a communication infrastructure", and page 38, Fig. 3.2., also second paragraph, and associated discussion respectively]; and
responding affirmatively to said query regardless of whether said server is able to respond to said message[See ORFALI, "send the method to the server", OMG, page 77, 2nd par. "object adaptor provides interface between ORB and object", IONA, page 7/23, e.g. "to aid remote operation", and page 38, "the OMG IDL skeleton program is the corresponding server-aid implementation of the OMG IDL interface"].

As to dependent claims 4 and 14:
See the rejection of independent claim 27.

As to dependent claim 5, and 15:
The method of claim 4 further comprising the step of creating an association between said client and said proxy object [ORFALI, page 4, item 5, "in the client by the local proxy objects", also first partial par., classes that provide the proxy client, server, and socket functions, OMG, page 74-75, IONA, page 9/23, 2nd to last par., "runtime builds a "proxy", and Mowbray et al., page 38].

As to dependent claims 6-8, and 16-18:
Please see the rejection of claim 27.

As to dependent claims 9, 19 and 29:
The method of claim 1 wherein said message includes an operation and a plurality of arguments, said step of generating further comprises the steps of:
translating said operation for said server;
translating said plurality of arguments for said server; and
generating a translated message using a message protocol of said Server [ORFALI, page 4, item 5, "the proxy object call cooperates with the local Objects Manager to find a server, marshals method argument, convert object pointers in the clients", and associated discussion; and OMG, page 78, 2nd paragraph, "object references are converted between stringfield and invocable forms only by the ORB", and associated discussion, also, IONA, page 7/23, 6th paragraph, "The IDL compiler is primarily a part of the development environment, and used to translate IDL description into sub code to aid remote operations", further, Mowbray, page 38, 2nd paragraph, Transparently, the stub provides an interface to the ORB that performs marshaling to encode and decode the operation's parameters into communication formats suitable for transmission", and associated discussion].

As to dependent claims 10-11, 20-21, and 30:
The method of claims 9 and 2, wherein said step of translating said arguments further comprises the steps of:
determining the expected number and type of arguments of said Server;
determining whether an expected argument type is different than an argument type; and
translating one of said plurality of arguments to an expected argument type when its type is different than said expected argument type [ORFALI, page 4, inherently teach of this step by Marshaling, see also OMG, the discussion of object reference and types on page 80, 2nd par.];

generating a translated response using a message protocol of said client [[ORFALI, page 4, item 5, "the proxy object call cooperates with the local Objects Manager to find a server, marshals method argument, convert object pointers in the clients", and associated discussion; and OMG, page 78, 2nd paragraph, "object references are converted between stringfield and invocable forms only by the ORB", and associated discussion, also, IONA, page 7/23, 6th paragraph, "The IDL compiler is primarily a part of the development environment, and used to translate IDL description into sub code to aid remote operations", further, Mowbray, page 38, 2nd paragraph, Transparently, the stub provides an interface to the ORB that performs marshaling to encode and decode the operation's parameters into communication formats suitable for transmission", and associated discussion]; and
transmitting, using said mapping, said translated response to said client [the transmitting of the translated message to the client, See ORFALI, "send the method to the server", OMG, page 77, 2nd par. "object adaptor provides interface between ORB and object", IONA, page 7/23, e.g. "to aid remote operation", and page 38, "the OMG IDL skeleton program is the corresponding server-aid implementation of the OMG IDL interface"].

As to dependent claims 23, and 28:
The message broker of claim 22 wherein said mediating component comprises:
a control module, said control module capable of creating a mapping between said client object and said server object [ORFALI, page2, "the object adaptor sits on top of ORB's core communication services and accept requests for service on behalf of the servers object", and page 1, last par., "to support both static and dynamic client/server invocations", OMG, page 77, 2nd par. IONA, page 11/23, Orbix

Architecture (see specifically 2nd par.), and Mowbray et al., page 36, and page 38, ORB];
a proxy object coupled to said a control module [ORFALI, page 4, item 5, "the proxy object call cooperates with the local Objects Manager to find a server, marshals method argument, convert object pointers in the clients", and associated discussion; and OMG, page 78, 2nd paragraph, "object references are converted between stringfield and invocable forms only by the ORB", and associated discussion, also, IONA, page 7/23, 6th paragraph, "The IDL compiler is primarily a part of the development environment, and used to translate IDL description into sub code to aid remote operations", further, Mowbray, page 38, 2nd paragraph, Transparently, the stub provides an interface to the ORB that performs marshaling to encode and decode the operation's parameters into communication formats suitable for transmission", and associated discussion]; and
a stub object coupled to said proxy object [ORFALI, page 1, "and both client and server stub are generated by the IDL compiler", OMG, page 87, IONA, page 9/23, 2nd to last par., "runtime builds a "proxy", and Mowbray et al., page 38].

As to dependent claim 24:
The message broker of claim 23 wherein said first object is a client object, and said proxy object is coupled to said client object [ORFALI, page 4, item 5, "the proxy object call cooperates with the local Objects Manager to find a server, marshals method argument, convert object pointers in the clients", and associated discussion; and OMG, page 78, 2nd paragraph, "object references are converted between stringfield and invocable forms only by the ORB", and associated discussion, also, IONA, page 7/23, 6th paragraph, "The IDL compiler is primarily a part of the development environment, and used to translate IDL description into sub code to aid remote operations", further, Mowbray, page 38, 2nd paragraph, Transparently, the stub

provides an interface to the ORB that performs marshaling to encode and decode the operation's parameters into communication formats suitable for transmission", and associated discussion].

As to dependent claim 25:
The message broker of claim 24 wherein said second object is a server object, and said stub object is coupled to said server object [ORFALI, page 1, "and both client and server stub are generated by the IDL compiler", OMG, page 87, IONA, page 9/23, 2nd to last par., "runtime builds a "proxy", and Mowbray et al., page 38].

As to dependent claim 26:
The message broker of claim 22 further comprising a second mediating component coupled to said mediating component [Mowbray, teaches of Skeleton, see also OMG, page 77, "there will be a skeleton for each object type bound to the ORB].

Prior Art not relied upon

The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

How to Contact the Examiner:

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Maijd Banankhah, whose telephone number is 571-272-3770. A voice mail service is also available at this number. The Examiner can normally be reached on Monday-Friday, except Tuesdays from 7:00 AM - 3:30 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, An Meng-AI who can be reached on 571-272-3756. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

All responses sent by U.S. Mail should be mailed to:

Commissioner for Patents
PO Box 1450
Alexandria, VA 22313-1450

PTO CENTRAL FAX NUMBER:
703-872-9306

- Any inquiry of a general nature or relating to the status of this application should be directed to the **TC 2100 Group receptionist: (571) 272-2100.**

MAJID BANANKHAH
PRIMARY EXAMINER